PATENT ABSTRACTS OF JAPAN

(11)Publication number : 06-067817
(43)Date of publication of application : 11.03.1994

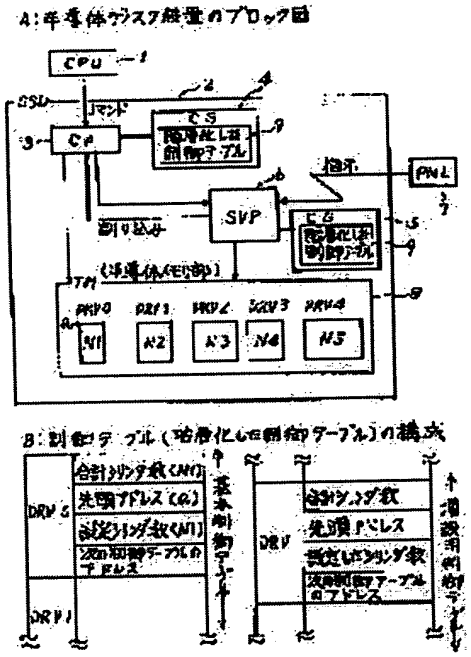
(51)Int.Cl. G06F 3/08
G06F 3/06

(21)Application number : 04-223827
(22)Date of filing : 24.08.1992

(71)Applicant : FUJITSU LTD
(72)Inventor : KURIHARA TAKUYA

(54) METHOD FOR MAINTAINING SEMICONDUCTOR DISK DEVICE

(57)Abstract:
PURPOSE: To improve the operation efficiency of a device by executing processing without turning off the device at the time of changing memory capac ity or repairing a fault memory.
CONSTITUTION: In the maintenance method of a semiconductor disk device 2 to be used by dividing a semiconductor memory part (TM) 8 into plural logical drives DRV0 to DRV1, hierarchical control tables 9 each of which consists of a basic control table and an extended control table are set up in the device 2 at every logical drive, and at the time of changing a memory, the control table 9 is updated at every logical drive, so that memory changing processing can be executed so as to make access logical drives other than the one concerned from a master device 1.



LEGAL STATUS

[Date of request for examination]	25.06.1996
[Date of sending the examiner's decision of rejection]	26.05.1998
[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]	
[Date of final disposal for application]	
[Patent number]	
[Date of registration]	
[Number of appeal against examiner's decision of rejection]	10-09954
[Date of requesting appeal against examiner's decision of rejection]	25.06.1998
[Date of extinction of right]	

(19)日本国特許庁（ＪＰ）　(12) **公開特許公報（A）**　(11)特許出願公開番号

特開平6－67817

(43)公開日　平成６年(1994)３月11日

(51)Int.Cl.[5]	識別記号	庁内整理番号	ＦＩ	技術表示箇所
Ｇ０６Ｆ　3/08		Ｈ　7165－5B		
3/06	３０１	Ｋ　7165－5B		

審査請求　未請求　請求項の数４(全 16 頁)

(21)出願番号　特願平4－223827

(22)出願日　平成４年(1992)８月24日

(71)出願人　000005223
富士通株式会社
神奈川県川崎市中原区上小田中1015番地
(72)発明者　栗原　拓弥
神奈川県横浜市港北区新横浜２丁目14番19号　株式会社富士通プログラム技研内
(74)代理人　弁理士　山谷　晧榮　（外１名）

(54)【発明の名称】　半導体ディスク装置の保守方法

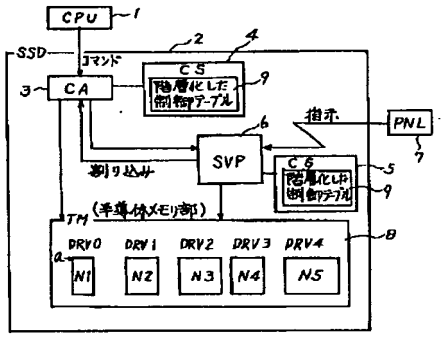
(57)【要約】

【目的】　本発明は半導体ディスク装置の保守方法に関し、メモリ容量の変更、或いは、障害メモリの修復時に、装置をオフラインにすることなく、処理が出来るようにして、装置の稼働効率を向上させることを目的とする。

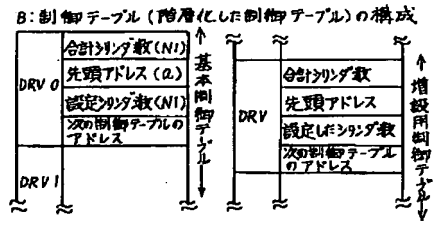
【構成】　半導体メモリ部（ＴＭ８）を、複数の論理ドライブＤＲＶ０、ＤＲＶ１、・・・に分割して使用する半導体ディスク装置２の保守方法において、半導体ディスク装置２の内部に、基本制御テーブルと、増設用制御テーブルとからなり、階層化した制御テーブル９を、各論理ドライブ毎に設定し、メモリの変更時に、各論理ドライブ毎に、制御テーブル９を更新処理することにより、該当する論理ドライブ以外の論理ドライブを、上位装置１からアクセス可能な状態で、メモリ変更処理を行うように構成した。

本発明の原理説明図

A：半導体ディスク装置のブロック図



B：制御テーブル（階層化した制御テーブル）の構成



1

【特許請求の範囲】

【請求項1】　記憶媒体として半導体メモリ（TM8）を使用し、

この半導体メモリ（TM8）を、所定の容量を持つ複数の論理ドライブ（DRV0、DRV1、・・・）に分割して使用することにより、

ディスク装置の入出力処理を仮想的に実行する半導体ディスク装置（2）の保守方法において、

該半導体ディスク装置（2）の内部に、上記論理ドライブの制御情報格納用の制御テーブル（9）を、各論理ドライブ（DRV0、DRV1、・・・）毎に設定すると共に、

この制御テーブル（9）を、階層化したテーブルとして設定し、

メモリの変更時に、各論理ドライブ（DRV0、DRV1、・・・）毎に、上記制御テーブル（9）を更新処理することにより、

該当する論理ドライブ以外の論理ドライブを、上位装置（1）からアクセス可能な状態で、メモリ変更処理を行うことを特徴とした半導体ディスク装置の保守方法。

【請求項2】　上記制御テーブル（9）を、

制御情報の初期設定値を格納する基本制御テーブルと、

該基本制御テーブルと同じ構成の増設用制御テーブルとで構成し、

これら基本制御テーブルと、増設用制御テーブルに、次の制御テーブルのアドレスを設定することにより、階層化したことを特徴とする請求項1記載の半導体ディスク装置の保守方法。

【請求項3】　オンラインで稼働中の上記半導体ディスク装置において、メモリを増設する際、

半導体ディスク装置（2）のプロセッサ（3、6）では、外部からの指示により、上記制御テーブル（9）を参照して、増設する論理ドライブの制御情報を獲得し、

増設する論理ドライブのみを、オフラインにして、増設部のフォーマット処理を行うと共に、

増設した論理ドライブの制御テーブル（9）に、制御情報の値を再定義し、

その後、増設部の論理ドライブをオンラインに戻すことを特徴とした請求項1記載の半導体ディスク装置の保守方法。

【請求項4】　オンラインで稼働中の上記半導体ディスク装置において、障害の発生したメモリの修復（交換）処理を行う際、

半導体ディスク装置（2）のプロセッサ（3、6）では、外部からの指示により、上記制御テーブル（9）を参照して、修復する論理ドライブの制御情報を獲得し、

修復する論理ドライブのみをオフラインにして、外部へ通知し、

その後、障害メモリの交換終了の指示を受けたプロセッサ（3、6）では、

2

上記制御テーブル（9）を参照して、オフラインにした論理ドライブのフォーマット処理を行い、

該フォーマット処理終了後、該当する論理ドライブを、オンラインに戻すことを特徴とした請求項1記載の半導体ディスク装置の保守方法。

【発明の詳細な説明】

【0001】

【産業上の利用分野】本発明は、記憶媒体として半導体メモリを使用し、磁気ディスク装置の入出力処理を、仮想的に実行する半導体ディスク装置の保守方法に関する。

【0002】

【従来の技術】図12～図14は、従来例の説明図であり、図12Aは半導体ディスク装置のブロック図、図12Bは制御テーブル例である。また、図13はシリンダ数を増やす場合の説明図であり、図13Aはフォーマット前、図13Bはフォーマット後を示す。更に、図14はメモリ障害の修復時の説明図である。

【0003】図12～図14中、1は中央処理装置（Central Processing Unit：以下「CPU」と記す）、2は半導体ディスク装置（SolidState Disk：以下「SSD」と記す）、3はチャネルアダプタ（Channel Adapter：以下「CA」と記す）、4、5はコントロールストレージ（Controll Strage：以下「CS」と記す）、6はサービスプロセッサ（以下「SVP」と記す）、7はメンテナンスパネル（以下「PNL」と記す）、8トラックメモリ（Track Memory：以下「TM」と記す）、9は制御テーブルを示す。

【0004】従来、記憶媒体に半導体メモリを使用し、ディスク装置の入出力処理を仮想的に実行する装置として、SSDが知られていた。その1例（例えば、特開昭61－165153号公報参照）を図12に示す。

【0005】①：SSDの構成の説明・・・図12参照図示のように、SSD2はCA3、SVP6、CS4、CS5、TM8、PNL7等で構成され、ホストシステムであるCPU1に接続して使用される。

【0006】このSSD2では、所定の容量を持つTM8（半導体メモリ部を構成）を、複数個の容量（例えば、N1～N5）に分割し、それぞれ独立した論理ドライブ（DRV0～DRV4）と見なすこと（エミュレート機能）により、データの入出力処理を行っている。

【0007】上記CA3はCPU1との間で、コマンド或いはデータのやりとり（入出力処理）や、SSD2のエミュレート機能をサポートするプロセッサであり、SVP6はSSD2の全体を制御するプロセッサである。

【0008】PNL7はオペレータ等により操作されるパネルであり、外部より、SVP6に対して指示を出し、保守を行わせるものである。CS4及びCS5内に

3

は、それぞれ、図１２Ｂに示したような制御テーブル９が設定されており、ＣＳ４内の制御テーブル９をＣＡ３が使用し、ＣＳ５内の制御テーブル９をＳＶＰ６が使用することにより、各種の制御（エミュレート）を行う。

【０００９】ＴＭ８は半導体メモリ部を構成するメモリであり、例えば、図示のように、容量の異なる複数の論理ドライブＤＲＶ０～ＤＲＶ４に分割して制御を行う。この場合、上記のように、ＣＳ４及びＣＳ５内に、制御テーブル９を設定し、各論理ドライブＤＲＶ０～ＤＲＶ４毎の容量（シリンダ数：ＣＬＹＮＵＭ）を登録しておく。

【００１０】この例では、論理ドライブＤＲＶ０～ＤＲＶ４に対し、その容量を、ＤＲＶ０＝Ｎ１、ＤＲＶ１＝Ｎ２、ＤＲＶ２＝Ｎ３、ＤＲＶ３＝Ｎ４、ＤＲＶ４＝Ｎ５のように設定する。

【００１１】②：論理ドライブの容量を変更（この例では増設）する場合の処理説明・・・図１２、図１３参照

上記のようにＴＭ８における論理ドライブＤＲＶ０～ＤＲＶ４の容量（シリンダ数）を、それぞれ、Ｎ１～Ｎ５に設定した状態で使用していたとする。この状態で、容量（シリンダ数）を増加する場合、次のようにする。

【００１２】②－１：増設するメモリを実装する。

②－２：ＣＰＵ１より、特定の命令をＳＳＤ２に対して発行する。この特定の命令により、全ての論理トライブのシリンダ数を指定する。

【００１３】②－３：上記命令により、ＣＡ３は、ＳＶＰ６に対して、制御テーブル９の設定変更を指示する。

②－４：ＳＶＰ６は、一旦配下の記憶領域であるＴＭ８内の各論理ドライブＤＲＶ０～ＤＲＶ４をオフラインにし、ＣＳ５内の制御テーブル９を、上記命令によって再定義する。

【００１４】②－５：上記処理が終了した時点で、ＳＶＰ６は、割り込みにより、ＣＡ３に、再定義処理の終了を通知する。

②－６：ＣＡ３は、上記割り込みを受け付けた後、ＣＳ４内の制御テーブル９を、上記命令に従って再定義し、制御テーブル９に基づいて、ＴＭ８の再フォーマットを行う。

【００１５】なお、上記のように、ＣＰＵ１より特定の命令を発行する代わりに、ＰＮＬ７より変更指示を出し、上記②－４～②－６の処理を行わせる方法もある。

以上の処理により、メモリの容量変更を行う。この場合、図１３に示したように、フォーマット前は、各論理ドライブＤＲＶ０～ＤＲＶ４の各容量（シリンダ数）は、ＤＲＶ０＝Ｎ１、ＤＲＶ１＝Ｎ１、ＤＲＶ２＝Ｎ２、ＤＲＶ３＝Ｎ４、ＤＲＶ４＝Ｎ５であり、これに、ＤＲＶ０＝Ｎ９、ＤＲＶ１＝Ｎ１０、ＤＲＶ２＝Ｎ１１の各容量を増設する。

【００１６】上記処理が終了したフォーマット後は、各論理ドライブＤＲＶ０～ＤＲＶ４の各容量（シリンダ

4

数）は、ＤＲＶ０＝Ｎ１＋Ｎ９、ＤＲＶ１＝Ｎ１＋Ｎ１０、ＤＲＶ２＝Ｎ２＋Ｎ１１、ＤＲＶ３＝Ｎ４、ＤＲＶ４＝Ｎ５となる。

【００１７】③：メモリ障害時の修復時の処理説明・・・図１２、図１４参照

ＴＭ８に障害が発生した場合、次のようにして処理を行う。

③－１：例えば、ＰＮＬ７からの指示により、ＳＶＰ６は、配下の論理ドライブＤＲＶ０～ＤＲＶ４をオフラインにする。

【００１８】③－２：障害の発生したメモリ部（例えば、ＤＲＶ２）を、良品と交換する。

③－３：ＳＶＰ６は、制御テーブル９をＣＳ５内に設定すると共に、この処理が終了した時点で、ＳＶＰ６は、割り込みにより、ＣＡ３に、再定義処理の終了を通知する。

【００１９】③－４：ＣＡ３は、上記割り込みを受け付けた後、ＣＳ４内の制御テーブル９を再定義し、制御テーブル９に基づいて、ＴＭ８の再フォーマットを行う。

【００２０】

【発明が解決しようとする課題】上記のような従来のものにおいては、次のような課題があった。

（１）、上記論理ドライブの容量を変更（シリンダ数の増設等）する場合、ＣＰＵ１又は、ＰＮＬ７からの設定情報を使用し、全ての論理ドライブのシリンダ数を設定／更新する。

【００２１】そのため、全てのドライブ（装置全体）を一度オフライン状態とし、増設するメモリと共に、それまで使用していたメモリを、ＣＡ３により、全て再フォーマット処理する。

【００２２】すなわち、１つの論理ドライブに付き、１つの制御テーブルで制御するような方法では、容量の変更を行う際、上記のように、ＳＳＤ２を全てＣＰＵ１から切り離した後、全てのメモリを再フォーマット化しなければならない。

【００２３】従って、ＳＳＤ２が連続して使用出来なくなるため、装置（ＳＳＤ）の稼働効率（利用効率）が低下する。

（２）、メモリ障害が発生した場合においても、上記容量変更時と同様に、ＳＳＤ２内の全ての論理ドライブを、一度オフライン状態にして、修復処理を行わなければならない。

【００２４】従って、この場合にも、装置（ＳＳＤ）の稼働効率（利用効率）が低下する。本発明は、このような従来の課題を解決し、メモリ容量の変更、或いは、障害メモリの修復時に、装置をオフラインにすることなく、処理が出来るようにして、装置の稼働効率（利用効率）を向上させることを目的とする。

【００２５】

【課題を解決するための手段】図１は本発明の原理説明

5

図であり、図１ＡはＳＳＤのブロック図、図１Ｂは制御テーブルの構成図である。図１中、図１２乃至図１４と同じものは、同一符号で示してある。

【００２６】本発明は上記の課題を解決するため、次のように構成した。

（１）、記憶媒体として半導体メモリ（ＴＭ８）を使用し、この半導体メモリ（ＴＭ８）を、所定の容量を持つ複数の論理ドライブＤＲＶ０、ＤＲＶ１、・・・に分割して使用することにより、ディスク装置の入出力処理を仮想的に実行する半導体ディスク装置２の保守方法において、該半導体ディスク装置２の内部に、上記論理ドライブの制御情報格納用の制御テーブル９を、各論理ドライブＤＲＶ０、ＤＲＶ１、・・・毎に設定すると共に、この制御テーブル９を、階層化したテーブルとして設定し、メモリの変更時に、各論理ドライブＤＲＶ０、ＤＲＶ１、・・・毎に、上記制御テーブル９を更新処理することにより、該当する論理ドライブ以外の論理ドライブを、上位装置１からアクセス可能な状態で、メモリ変更処理を行うようにした。

【００２７】（２）、上記構成（１）において、制御テーブル９を、制御情報の初期設定値を格納する基本制御テーブルと、該基本制御テーブルと同じ構成の増設用制御テーブルとで構成し、これら基本制御テーブルと、増設用制御テーブルに、次の制御テーブルのアドレスを設定することにより、階層化した。

【００２８】（３）、上記構成（１）において、オンラインで稼働中の上記半導体ディスク装置において、メモリを増設する際、半導体ディスク装置２のプロセッサ（ＣＡ３、ＳＶＰ６）では、外部からの指示により、上記制御テーブル９を参照して、増設する論理ドライブの制御情報を獲得し、増設する論理ドライブのみを、オフラインにして、増設部のフォーマット処理を行うと共に、増設した論理ドライブの制御テーブル９に、制御情報の値を再定義し、その後、増設部の論理ドライブをオンラインに戻すようにした。

【００２９】（４）、上記構成（１）において、オンラインで稼働中の上記半導体ディスク装置において、障害の発生したメモリの修復（交換）処理を行う際、半導体ディスク装置２のプロセッサ（ＣＡ３、ＳＶＰ６）では、外部からの指示により、上記制御テーブル９を参照して、修復する論理ドライブの制御情報を獲得し、修復する論理ドライブのみをオフラインにして、外部へ通知し、その後、障害メモリの交換終了の指示を受けたプロセッサ（ＣＡ３、ＳＶＰ６）では、上記制御テーブル９を参照して、オフラインにした論理ドライブのフォーマット処理を行い、該フォーマット処理終了後、該当する論理ドライブを、オンラインに戻すようにした。

【００３０】

【作用】上記構成に基づく本発明の作用を、図１に基づいて説明する。各論理ドライブに対し、基本制御テーブ

6

ルを図１Ｂのように設定し、この基本制御テーブルと同じ構成のテーブルを、増設用制御テーブルとして設定する。増設用制御テーブルは、ドライブ、或いはシリンダを増設する時に使用する。

【００３１】そして、制御テーブルに「次の制御テーブルのアドレス」の項を設け、この項に、次のテーブルの先頭アドレスを設定出来るようにして、該制御テーブルを階層化する。

【００３２】また、基本制御テーブルには、「合計シリンダ数」（例えばＮ１）が設定出来るようになっており、この値によりそのドライブの有無を決めることが出来る。このようにして階層化した制御テーブル９により、論理ドライブに属するメモリ（ＴＭ８）の範囲を動的に現す事が可能となる。

【００３３】上記のような階層化された制御テーブル９を設定した半導体ディスク装置２において、メモリの増設時等に、例えば、ＳＶＰ６により、上記制御テーブル９が更新されると、ＳＶＰ６からＣＡ３へ割り込みを行う。

【００３４】ＣＡ３ではこの割り込み処理において、制御テーブルが更新されたことを認識し、更新された制御テーブル９の制御情報を用いて、更新対象メモリ（ＴＭ８）部のフォーマット処理を行う。

【００３５】この場合、更新対象の論理ドライブのみをオフラインとし、更新対象外の論理ドライブは、オンライン状態で、上記処理を行う。また、メモリ障害時の修復処理時においても、上記と同様に、修復する論理ドライブのみ、オフラインとし、その他の論理ドライブは、オンラインのまま、修復処理を行う事が出来る。

【００３６】以上の通り、メモリの変更（増設等）の際に、装置をオンライン状態にしたまま、該当する論理ドライブのみをオフラインにして、処理を行うことが出来る。従って、装置を連続使用出来るから、装置の稼働効率（利用効率）が向上し、処理時間も短縮出来る。

【００３７】

【実施例】以下、本発明の実施例を図面に基づいて説明する。図２～図１１は、本発明の実施例を示した図であり、図２は半導体ディスク装置のブロック図、図３は基本制御テーブルとＴＭの関係例を示した図、図４は増設後の制御テーブルの構成例及びＴＭの関係例を示した図、図５は実施例１（ドライブを増やす場合）の制御テーブル例を示した図、図６は実施例１のＴＭの構成例を示した図、図７はＳＶＰより増設指示を出して増設する場合の処理フローチャート、図８はＳＶＰより指示を出して増設する場合の処理フローチャート、図９は実施例２（シリンダを増やす場合）の制御テーブル例を示した図、図１０は実施例２の説明図（ＴＭの構成例）、図１１はメモリ障害時の修復（交換）を行う場合の処理フローチャートである。

【００３８】図２～図１１中、図１、及び図１２乃至図

7

１４と同じものは、同一符号で示してある。また、１０は比較用制御テーブル（制御テーブル９と同じ構成）を示す。

【００３９】①：ＳＳＤ（半導体ディスク装置）の構成の説明・・・図２参照

図示のように、ＳＳＤ２は、ＣＡ３、ＳＶＰ６、ＣＳ４、ＣＳ５、ＴＭ８、ＰＮＬ７等で構成され、ホストシステムであるＣＰＵ１に接続して使用される。

【００４０】そして、上記ＣＳ４には制御テーブル９を設定し、ＣＳ５には制御テーブル９と、比較用制御テーブル１０とを設定する。この比較用テーブル１０は、制御テーブル９と同じ構成であり、外部（ＰＮＬ７等）からの制御情報に基づいてＳＶＰ６が値をセットする。なお、ＣＳ５以外の構成は、上記従来例と同じなので、説明は省略する。

【００４１】②：基本制御テーブルと、ＴＭ８の関係例の説明・・・図３参照

本実施例では、各論理ドライブに対し、各制御テーブル９に、図３に示したような基本制御テーブル（最初に設定した制御テーブル）を設定する。また、各制御テーブル９に、基本制御テーブルと同じ構成のテーブル（増設用テーブル）を、メモリの増設用として設定する。

【００４２】図３に示した制御テーブル９の各項目は、次の通りである。すなわち、「Ｔｏｔａｌ　ＣＹＬＮＵＭ」は、当該論理ドライブの合計のシリンダ数（増設する度に変更）（Ｎ１、Ｎ２、・・）を示し、「ＴｏｐＡｄｒｓ－Ｂａｓｉｃ」は当該論理ドライブの先頭アドレス（最初に設定した数値のまま）｛（ａ）、（ｂ）、・・・｝を示す。

【００４３】また、「Ｂａｓｉｃ　ＣＹＬＮＵＭ」は、当該制御テーブル９で設定するシリンダ数（増設テーブルでは、増設するシリンダ数）（Ｎ１、Ｎ２、・・）を示し、「ＡＤＤ　Ｐｏｉｎｔｅｒ」は当該論理ドライブの制御テーブル９の次の制御テーブルのアドレスを示す。なお、制御テーブル９が、基本制御テーブルのみの場合、及び当該ドライブの最終増設テーブルは、「０」とする。

【００４４】上記のように、増設用テーブルは、シリンダ数を増やす時に使用し、制御テーブル９の「ＡＤＤ　Ｐｏｉｎｔｅｒ」に、その制御テーブルの次の制御テーブルのアドレスを示し、「ＡＤＤ　Ｐｏｉｎｔｅｒ」が「０」の時は、そのテーブルが最終制御テーブルとなる。そして、この「ＡＤＤ　Ｐｏｉｎｔｅｒ」を使い、制御テーブル９を階層化する。

【００４５】また、各ドライブの基本制御テーブルの「Ｔｏｔａｌ　ＣＹＬＮＵＭ」によって、その論理ドライブの有無を決める（その値が「０」の時は、その論理ドライブは構成しない）。これにより、論理ドライブＤＲＶ０～ＤＲＶ４に属するＴＭ８の範囲を動的に現せる。

8

【００４６】この制御テーブル９の改版が行われた時は、ＣＡ３に対する割り込み処理において、ＣＡ３が認識し、更新された制御テーブル９のフォーマットにより、更新対象のＴＭ８を再フォーマット処理する。この場合、更新対象外の論理ドライブは、オンラインのまま、処理を続けることが出来る。

【００４７】③：増設後の制御テーブルの構成及び、ＴＭの関係例の説明・・・図４参照

上記図３に示した制御テーブルを増設した場合の制御テーブル及び、ＴＭ８の構成例を図４に示す。

【００４８】この例では、論理ドライブＤＲＶ０に、Ｍ１の容量（シリンダ数）を増設し、論理ドライブＤＲＶ２を新しく増設した例である。この場合、増設前（図３の状態）は、ＤＲＶ０のシリンダ数はＮ１、ＤＲＶ１のシリンダ数はＮ２、ＤＲＶ２のシリンダ数はＮ３＝０、ＤＲＶ３のシリンダ数はＮ４＝０である。

【００４９】そして、増設後は（図４の状態）、ＤＲＶ０のシリンダ数はＮ１＋Ｍ１（Ｍ１：増設分）、ＤＲＶ１のシリンダ数はＮ２、ＤＲＶ２のシリンダ数はＮ３（０でない数）、ＤＲＶ３のシリンダ数はＮ４＝０となる。

【００５０】従って、ＤＲＶ０の「Ｔｏｔａｌ　ＣＹＬＮＵＭ」は、Ｍ１を増設するので、Ｎ１＋Ｍ１の値を設定する。また、論理ドライブＤＲＶ０の基本制御テーブルの「ＡＤＤ　Ｐｏｉｎｔｅｒ」は増設テーブルの先頭アドレスを示しており、論理ドライブＤＲＶ０の基本制御テーブル以外の「ＡＤＤ　Ｐｏｉｎｔｅｒ」は、次のテーブルが無いために、「０」を設定する。

【００５１】なお、当該ドライブが存在しない場合は、基本制御テーブルの「ＴｏｔａｌＣＹＬＮＵＭ」に「０」を設定する。ところで、コンピュータシステムにおける保守のニーズとして、活性保守（装置の電源を落とさずに、部品交換を行い、システム運転を停止させないで、保守する）が要求されている中で、例えば、電源投入のまま、プリント板を挿抜する手法が開発されている。

【００５２】本実施例では、上記の手法を利用して、ＳＳＤ内のメモリプリント板（論理ドライブ）の増設削除、或いは、修復を行う。また、上記メモリの増設方法には、論理ドライブ数を増設する方法と、各論理ドライブのシリンダ数を増設する場合と、これら２つを組み合わせた方法との３つの方法がある。

【００５３】以下、具体的な実施例について、詳細に説明する。

④：実施例１（ドライブ数を増設する方法）の説明・・・図５、図６、図７、図８参照

④－１：制御テーブル及び、ＴＭに基づく説明・・・図５、図６参照

この例は、最大ドライブ数８のＳＳＤにおいて、４ＫＢｙｔｅのメモリを、通常ドライブ数４（ＤＲＶ０～ＤＲ

9

V3）、1シリンダ当たり、0x80Byte（128Byte）で、各ドライブに、DRV0－4シリンダ、DRV1－6シリンダ、DRV2－12シリンダ、DRV3－10シリンダ、の条件で使用していたものに、2KByteのメモリを増設し、DRV4－6シリンダ、DRV5－2シリンダ、DRV6－8シリンダ、の3つのドライブを増設する例である。

【0054】図5に示した制御テーブル例において、ドライブDRV0、DRV1、DRV2、DRV3が今まで使用していた論理ドライブ（増設前のドライブ）であり、ドライブDRV4、DRV5、DRV6が増設する論理ドライブである。

【0055】ドライブDRV0～DRV3の制御テーブルにおいて、「Top AdrsBasic」或いは、「Basic CYLNUM」に変更が有った場合は、論理ドライブの設定のやり直しと判断し、SVP6において、そのことを判断させ、従来と同じ処理で論理ドライブを設定する。

【0056】上記処理時のTM8の構成を図6に示す。図示のように、増設後のTM8には、DRV4、DRV5、DRV6が増設されている。なお、この判断は、CA3の割り込み処理においても実行させる事が出来るが、SVP6において、処理したほうが実行処理が少なくなるので、この例では、SVP6で判断させる。

【0057】また、DRV0～DRV3の制御テーブルにおいて、「Top Adrs Basic」或いは、「Basic CYLNUM」以外の所に変更がある場合は、後述する。

【0058】④－2：フローチャートに基づく説明・・・図7、図8参照

図7、図8は、SVPより指示を出して、増設する場合の処理フローチャートであり、以下、図7、図8に基づいて説明する。なお、図のS1～S14は、各処理番号を示す。

【0059】（S1）：先ず、SVP6は、外部からの制御情報に基づき、CS5内の比較用制御テーブル10に値をセットする。

（S2）：SVP6より増設を指示する。

【0060】（S3）：SVP6により、今設定されている（オンライン中の）制御テーブル9と、比較用制御テーブル10とを比べ、増設部以外に変更点は無いかを判断する。もし、変更点があった場合は、従来例（上記従来例の②－4からの処理）と同じように処理する。

【0061】（S4）：しかし、変更点があった場合には、SVP6により、比較用制御テーブル10に、正常な値がセットされているか（アドレスの重複が無いか等）を確認する。もし、重複があれば、制御テーブルの再定義を促す。

【0062】（S5）：比較用制御テーブル10に、正常な値がセットされている場合には、SVP6より、CA3に対して、割り込みを行う。

10

（S6）：この時、SVP6では、CS5内の制御テーブル9に値（制御情報の値）を再定義し、通常運用を行う。

【0063】（S7）：CA3は、上記割り込みにより、各ドライブに対して、以下の処理を実行する。

（S8）：CA3では、基本ドライブ分（最大ドライブ数）処理を行ったかどうかを調べ、処理が完了していれば、通常運行を行う。

【0064】（S9）：しかし、処理が完了していなければ、CA3は、今設定されている（オンライン中の）制御テーブル9と、これから設定する制御テーブルとを比べ、どのように（ドライブを幾つ、シリンダが幾つ）フォーマットするか調べる。

【0065】すなわち、制御テーブルに変更があるかを調べ、無ければ、次のドライブの処理に移る。

（S10）：CA3は、制御テーブル9内の「Total CYLNUM」が「0」かを調べる。

【0066】（S11）：もし、「Total CYLNUM」が「0」ならば、当該ドライブを削除（使用しない）するので、当該ドライブを、オフラインにする。

（S12）：そして、CS4内の制御テーブル9に値を再定義する。

【0067】（S13）：また、「Total CYLNUM」が「0」でない場合は、増設部のみオフラインにして、該増設部にのみ、フォーマット処理を行う。

（S14）：終了したら、増設したドライブに対してCA3がエミュレートを行えるように、CS4内の制御テーブル9に値を再定義する。

【0068】（S15）：CA3は増設したドライブをオンラインにする。

以上の各処理により、増設部のみフォーマットして、他はオンラインのままで、メモリの増設を行うことが出来る。

【0069】なお、この例では、3つのドライブを増設したが、制御テーブルの基本テーブルの数を変えることにより、増設出来るドライブの数を設定する。また、増設出来るドライブ数を越えなければ、何度でも増設が可能である。

【0070】⑤：実施例2（各ドライブのシリンダ数を増設する例）の説明・・・図9、図10参照

図9はシリンダを増設する場合の制御テーブルの例である。また、図10は、実施例2におけるTM8の構成例であり、図10Aは増設前のTM8の構成例、図10Bは増設後のTM8の構成例、図10CはCA3から見たTM8の考え方（エミュレートの仕方）を示す。

【0071】⑤－1：制御テーブル及び、TMによる説明・・・図9、図10参照

この例は、最大ドライブ数5のSSDにおいて、4KByteのメモリを、通常ドライブ数4（DRV0～DR

11

Ｖ３）、１シリンダ当たり、０ｘ８０Ｂｙｔｅ（１２８Ｂｙｔｅ）で、各ドライブに、ＤＲＶ０－４シリンダ、ＤＲＶ１－６シリンダ、ＤＲＶ２－１２シリンダ、ＤＲＶ３－１０シリンダ、の条件で使用していたものに、２ＫＢｙｔｅのメモリを増設し、ＤＲＶ０、ＤＲＶ１、ＤＲＶ３の３つのドライブに、それぞれ、６、２、８シリンダづつ増設する例である。

【００７２】図示のように、ＤＲＶ０～ＤＲＶ３までが、今まで使用していたドライブであり、そのうち、ＤＲＶ０、ＤＲＶ１、ＤＲＶ３にそれぞれシリンダを増設する。ＤＲＶ０～ＤＲＶ３の制御テーブルにおいて、「Ｔｏｐ　Ａｄｓ　Ｂａｓｉｃ」或いは「Ｂａｓｉｃ　ＣＹＬＮＵＭ」に変更が有った場合は、論理ドライブの設定のやり直しと判断し、ＳＶＰ６において、そのことを判断させ、従来と同じ処理において、論理ドライブを設定する（理由は上記実施例１と同じ）。

【００７３】⑤－２：フローチャートに基づく説明・・・図７、図８参照

上記図７、図８は、実施例２でも共通しているので、以下、図７、図８に基づいて、実施例２の処理を説明する。なお、図のＳ１～Ｓ１４は、各処理番号を示す。

【００７４】（Ｓ１）：先ず、ＳＶＰ６は、外部からの制御情報に基づいて、ＣＳ５内の比較用制御テーブル１０に値をセットする。

（Ｓ２）：ＳＶＰ６より増設を指示する。

【００７５】（Ｓ３）：ＳＶＰ６により、今設定されている（オンライン中の）制御テーブル９と、比較用制御テーブル１０とを比べ、増設部以外に変更点は無いかを判断する。もし、変更点があった場合は、従来例と同じように処理する。

【００７６】（Ｓ４）：しかし、変更点がなければ、ＳＶＰ６により、比較用制御テーブル１０に、正常な値がセットされているか（アドレスの重複が無いか等）を確認する。もし、重複があれば、制御テーブルの再定義を促す。

【００７７】（Ｓ５）：比較用制御テーブル１０に、正常な値がセットされている場合には、ＳＶＰ６より、ＣＡ３に対して、割り込みを行う。

（Ｓ６）：この時、ＳＶＰ６では、ＣＳ５内の制御テーブル９に値を再定義し、通常運用を行う。

【００７８】（Ｓ７）：ＣＡ３は、上記割り込みにより、各ドライブに対して、以下の処理を実行する。

（Ｓ８）：ＣＡ３では、基本ドライブ分（最大ドライブ数）処理を行ったかどうかを調べ、処理が完了していれば、通常運用を行う。

【００７９】（Ｓ９）：しかし、処理が完了していなければ、今設定されている（オンライン中の）制御テーブル９と、これから設定する制御テーブルとを比べ、どのように（幾つシリンダを増設するか）フォーマットするか調べる。

12

【００８０】この時、制御テーブル９の「ＡＤＤ　Ｐｏｉｎｔｅｒ」を調べ、「０」でない時は、階層的に次のテーブルを調べていき、「０」になるまで調べる。そして、制御テーブルに変更があるかを調べ、なければ、次のドライブの処理に移る。

【００８１】（Ｓ１０）：ＣＡ３は、制御テーブル９内の「Ｔｏｔａｌ　ＣＹＬＮＵＭ」が「０」かを調べる。

（Ｓ１１）：もし、「Ｔｏｔａｌ　ＣＹＬＮＵＭ」が「０」ならば、当該ドライブを削除（使用しない）するので、当該ドライブを、オフラインにする。

【００８２】（Ｓ１２）：そして、ＣＳ４内の制御テーブル９に値を再定義する。

（Ｓ１３）：また、「Ｔｏｔａｌ　ＣＹＬＮＵＭ」が「０」でない場合は、増設部のみオフラインにして、該増設部にのみ、フォーマット処理を行う。

【００８３】（Ｓ１４）：終了したら、増設したドライブに対してＣＡ３がエミュレートを行えるように、ＣＳ４内の制御テーブル９に値を再定義する。

（Ｓ１５）：増設したドライブをオンラインにする。

【００８４】以上の各処理により、増設部のみフォーマットして、他はオンラインのままで、メモリの増設を行うことが出来る。なお、この例では、増設は１回だったが、「ＡＤＤ　Ｐｏｉｎｔｅｒ」を次々に指定することにより、何段階にも、階層的に制御テーブルを繋げることが出来、それにより、何度でも、増設が可能である。

【００８５】なお、上記（Ｓ１４）の処理において、実際は、図１０Ａ、図１０ＢのようにＴＭ８は構成されるが、ＣＳ４内の制御テーブル９を用いてエミュレートする際、ＣＡ３は、図１０ＣのようにＴＭ８を見る。

【００８６】例えば、ＤＲＶ０を例にすれば、ＤＲＶ０の０ｘ２２０Ｂｙｔｅ目より、アクセスしようとする場合、ＣＡ３によって、０ｘ１０２０番地よりアクセスするようにエミュレートする。

【００８７】

⑥：実施例３（上記２つの方法を合わせた方法）

この例は、上記２つの方法、すなわち、ドライブの増設と、シリンダの増設とを合わせて行う例である。

【００８８】実施例３は、上記実施例２の制御テーブルを複数個余分に持つことによって、実現出来る。すなわち、上記実施例１と実施例２を合わせた処理なので説明は省略する。

【００８９】⑦：実施例４（メモリ障害時にメモリを修復する場合の実施例）・・・図１１参照

この例は、ＴＭ８の或るアドレスにおいて、メモリ障害が発生した時の修復処理の例である。以下、図１１の処理フローチャートに基づいて、実施例４の処理を説明する。なお、図１１のＳ２１～Ｓ３１は、各処理番号を示す。

【００９０】（Ｓ２１）：ＰＮＬ７より障害メモリのアドレスを指定して、修復処理を行うことをＳＶＰ６へ指

13

示する（例えば、カスタマエンジニアが指示する）。

（Ｓ２２）：ＳＶＰ６はＣＳ５内の制御テーブル９を調べ、障害メモリのアドレスが、どのドライブに割り振られているかを調べる。

【００９１】（Ｓ２３）：ＳＶＰ６は、ＣＡ３に対して、上記（Ｓ２２）で調べたドライブをオフラインにする割り込みを行う。

（Ｓ２４）：ＣＡ３の割り込み処理において、指示されたドライブをオフラインにする。

【００９２】（Ｓ２５）：ＣＡ３は、オフラインになったことを、ＳＶＰ６に通知する。

（Ｓ２６）：ＳＶＰ６は、オフラインになったことを確認したら、ＰＮＬ７を通して、ＰＮＬ７を操作している人（例えば、カスタマエンジニア）へ知らせる。

【００９３】（Ｓ２７）：例えば、カスタマエンジニアは、障害メモリを新品と交換する。

（Ｓ２８）：例えば、カスタマエンジニアは、ＰＮＬ７より、オフラインにしたドライブを、オンラインにするように、ＳＶＰ６へ指示する（交換終了指示）。

【００９４】（Ｓ２９）：ＳＶＰ６は、ドライブをオンラインにするように、ＣＡ３へ割り込みを行う。

（Ｓ３０）：ＣＡ３では、割り込み処理において、オフラインにしたドライブの制御テーブル９（ＣＳ４内の制御テーブル）を、参照して、ドライブのフォーマット処理を行う。

【００９５】（Ｓ３１）：その後、ＣＡ３は、フォーマット処理したドライブを、オンラインにする。

以上の各処理により、障害メモリ部のドライブのみオフラインにするだけで、障害メモリの修復を行う事が可能となる。

【００９６】

【発明の効果】以上説明したように、本発明によれば次のような効果がある。

（１）、装置をオフラインにすることなく、メモリを増設したり、或いは障害メモリの修復をしたりする事が出来る。従って、半導体ディスク装置の連続使用が出来、該装置の稼働効率（利用効率）が向上する。

14

【００９７】（２）、装置をオフラインにすることなく、メモリを増設したり、或いは障害メモリの修復をしたりする事が出来る。従って、メモリの増設時、或いは障害メモリの修復時の処理時間が短縮できる。

【図面の簡単な説明】

【図１】本発明の原理説明図である。

【図２】本発明の実施例における半導体ディスク装置のブロック図である。

【図３】本発明の実施例における基本制御テーブルとＴＭの関係例である。

【図４】本発明の実施例における増設後の制御テーブルの構成例及びＴＭの関係例である。

【図５】実施例１（ドライブを増やす場合）の制御テーブル例である。

【図６】実施例１のＴＭの構成例である。

【図７】ＳＶＰより増設指示を出して増設する場合の処理フローチャート（その１：ＳＶＰの処理）である。

【図８】ＳＶＰより増設指示を出して増設する場合の処理フローチャート（その２：ＣＡの処理）である。

【図９】実施例２（シリンダを増やす場合）の制御テーブル例である。

【図１０】実施例２の説明図（ＴＭの構成例）である。

【図１１】メモリ障害時のメモリの修復（交換）を行う場合の処理フローチャートである。

【図１２】従来例の説明図（その１）である。

【図１３】従来例の説明図（その２）である。

【図１４】従来例の説明図（その３）である。

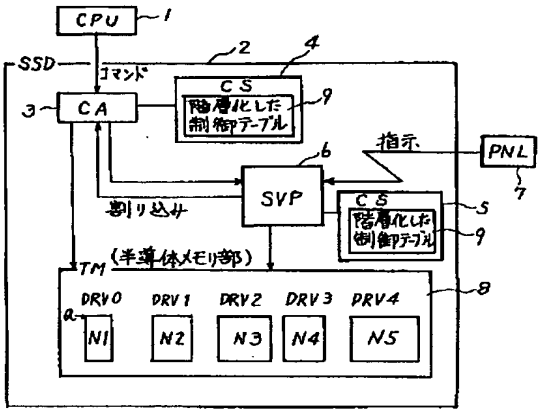
【符号の説明】

１　ＣＰＵ（中央処理装置）
２　ＳＳＤ（半導体ディスク装置）
３　ＣＡ（チャネルアダプタ）
４、５　ＣＳ（コントロールストレージ）
６　ＳＶＰ（サービスプロセッサ）
７　ＰＮＬ（メンタナンスパネル）
８　ＴＭ（トラックメモリ）
９　制御テーブル

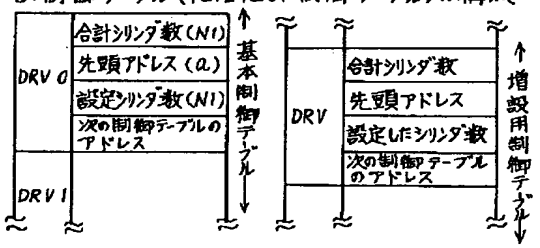
【図1】

本発明の原理説明図

A：半導体ディスク装置のブロック図

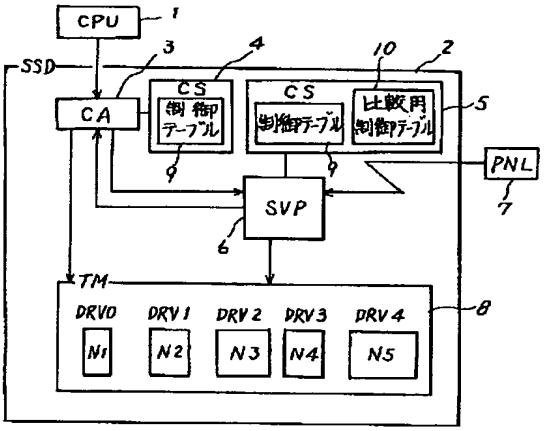


B：制御テーブル（階層化した制御テーブル）の構成



【図2】

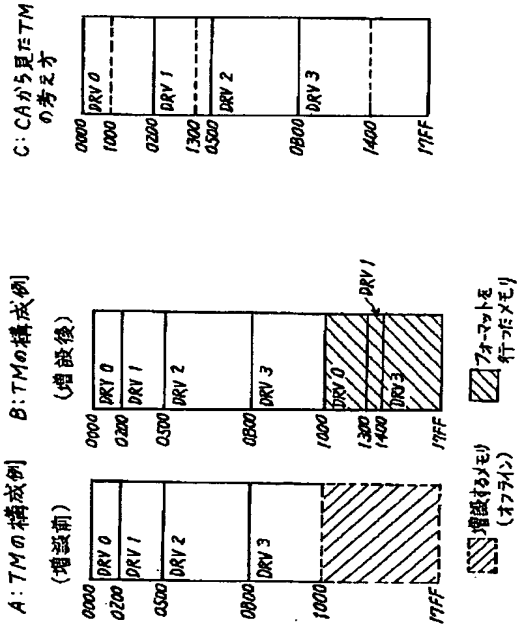
実施例における半導体ディスク装置のブロック図



【図10】

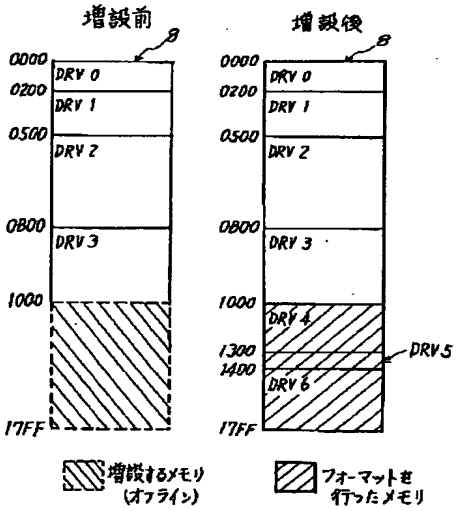
実施例2の説明図

(TMの構成例)



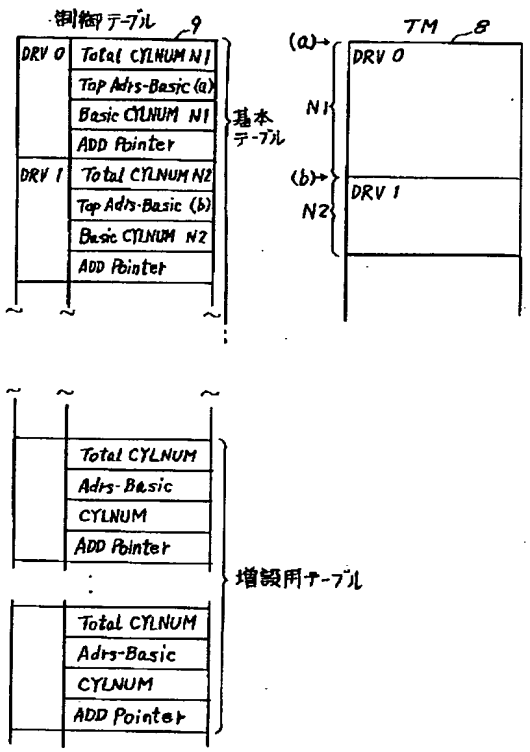
【図6】

実施例1のTMの構成例



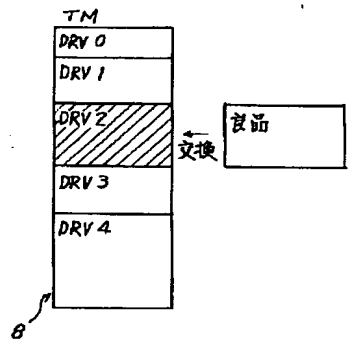
【図３】

基本制御テーブルとTMの関係例



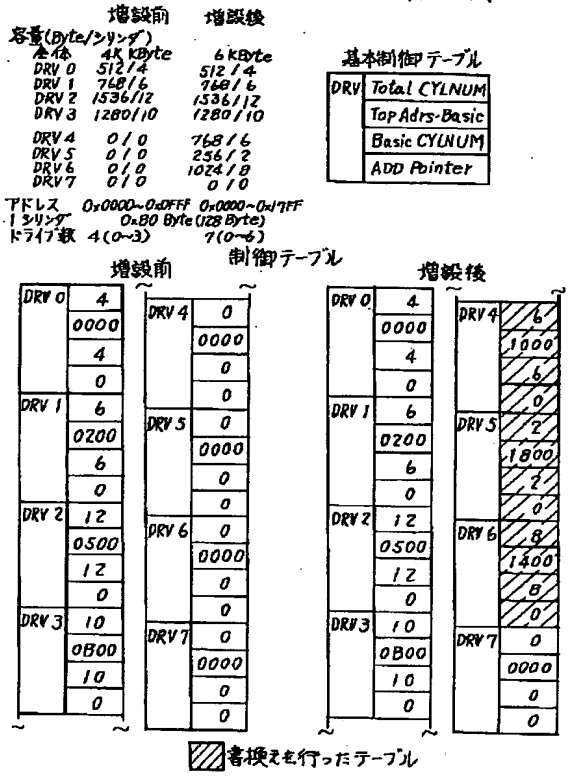
【図１４】

従来例の説明図（その３）
（メモリ障害の修復時）



【図５】

実施例１：（ドライブを増やす場合）の制御テーブル例



【図４】

増設後の制御テーブルの構成例及びTMの関係例

最大ドライブ数　４ドライブ

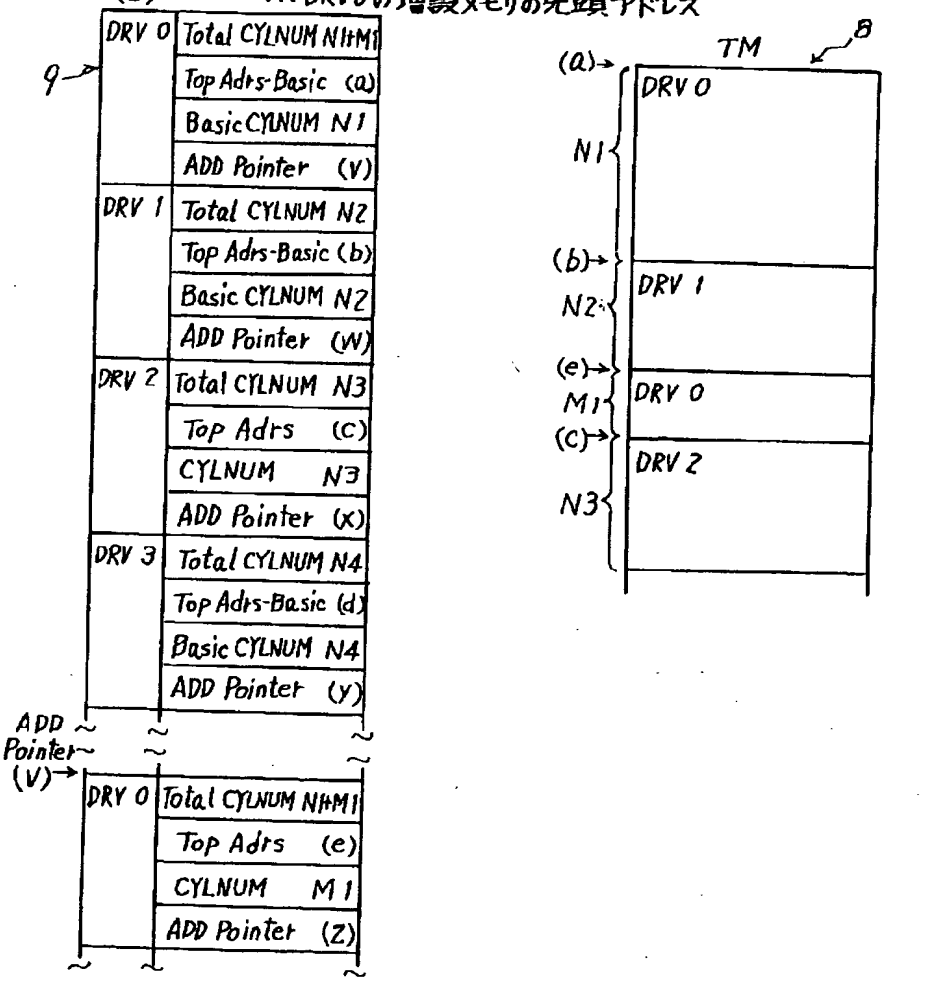
増設前（図３参照）

DRV 0　シリンダ数 … N1
1 … N2
2 … N3=0
3 … N4=0

増設後

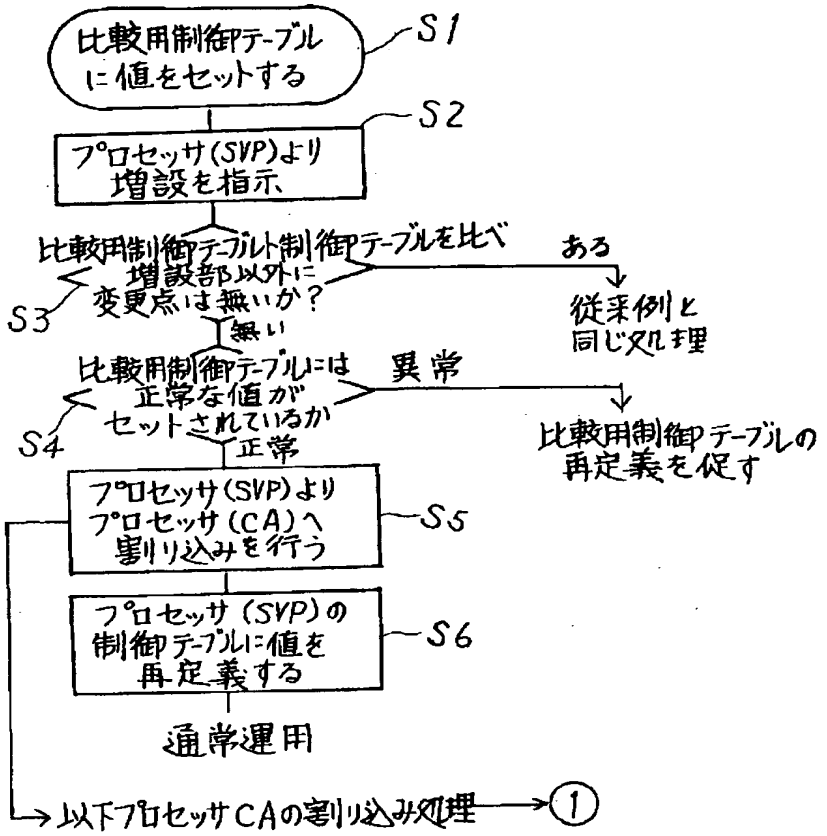
DRV 0　シリンダ数 … N1+M1
1 … N2
2 … N3≠0
3 … N4=0

N1～N4.M1 … それぞれのシリンダ数
(a)～(d) … それぞれのドライブの先頭アドレス
(V)～(z) … それぞれの増設テーブルの先頭アドレス
（この例では(W)～(z)の値は"0"とする．(V)のみ値をセットする）
(e) … DRV0の増設メモリの先頭アドレス



【図7】

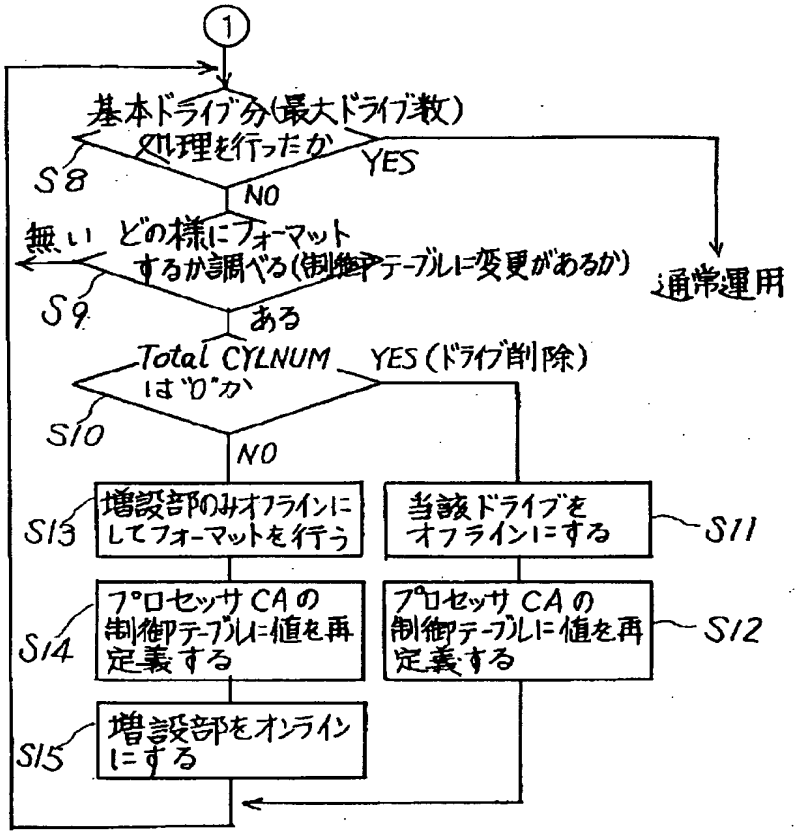
SVPより増設指示を出して増設する場合の処理フローチャート（その1：SVPの処理）



（S1～S6：SVPの処理）

【図8】

SVPより増設指示を出して増設する場合の処理フローチャート（その2：CAの処理）



（S8～S15：CAの処理）

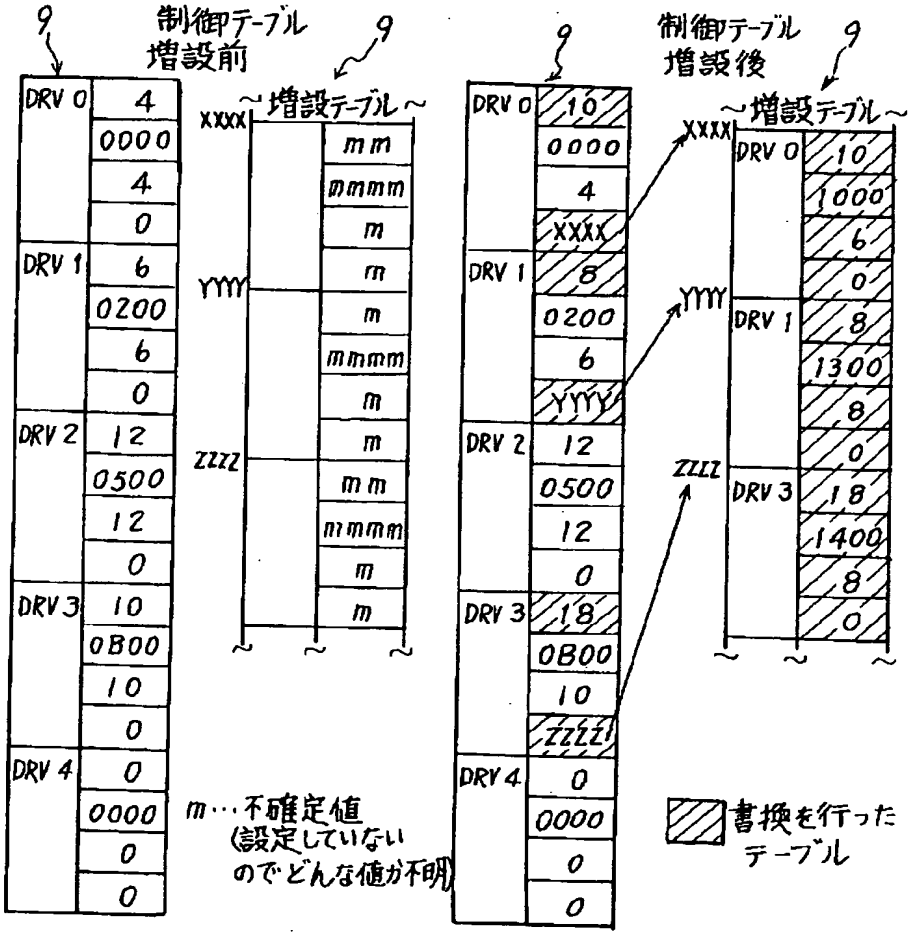
【図9】

実施例2:(シリンダを増やす場合)の制御テーブル例

	増設前	増設後
容量(Byte/シリンダ)		
全体	4 KByte	6 KByte
DRV0	512/4	1280/10
DRV1	768/6	1024/8
DRV2	1536/12	1536/12
DRV3	1280/10	2304/18
DRV4	0/0	0/0
アドレス	0x0000~0x0FFF	0x0000~0x17FF
1シリンダ	0x80 Byte(128 Byte)	
ドライブ数	4(0~3)	4(0~3)

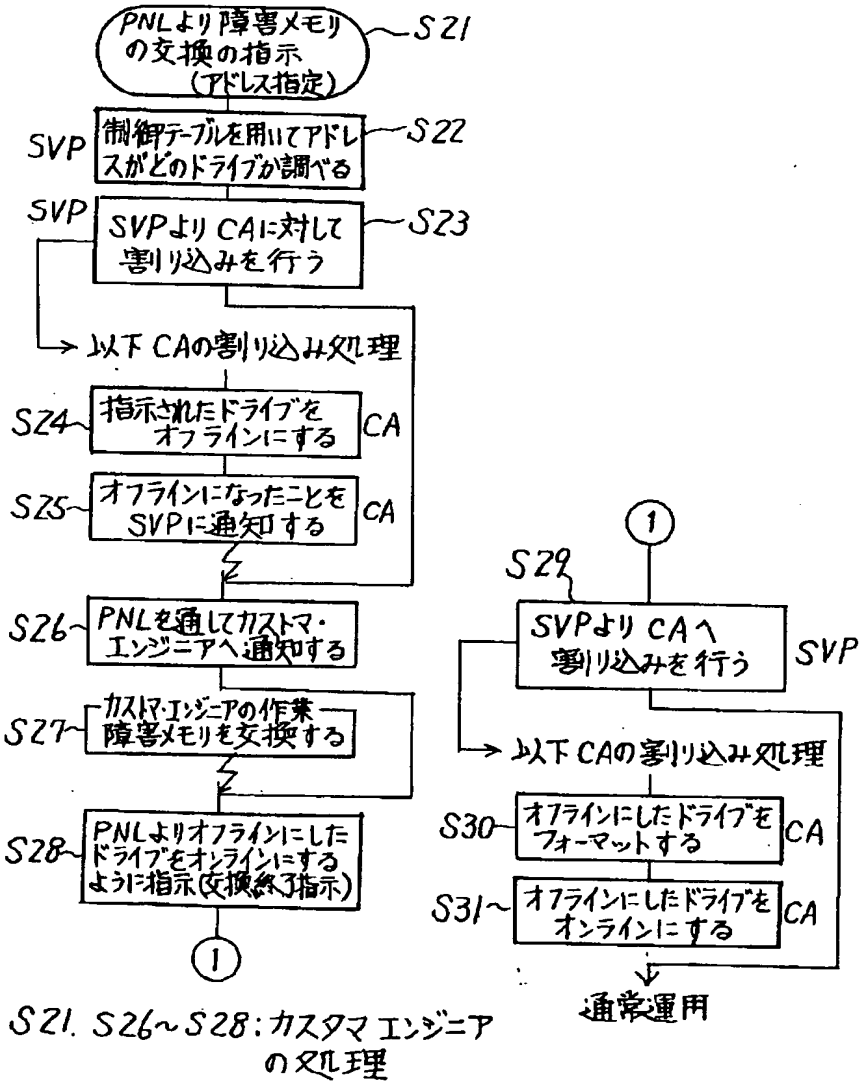
基本制御テーブル

DRV	Total CYLNUM
	Top Adrs-Basic
	Basic CYLNUM
	ADD Pointer



【図11】

メモリ障害時のメモリの修復(交換)を行う場合の処理フローチャート

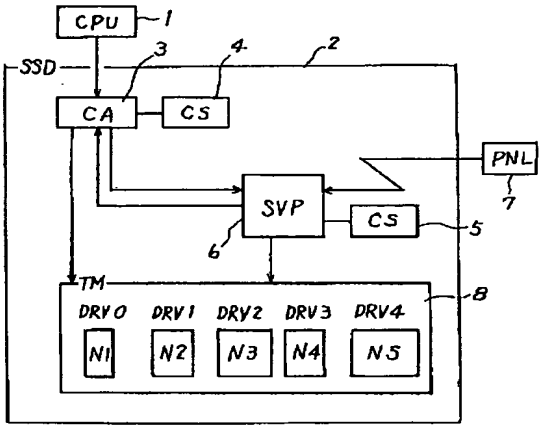


S21. S26～S28：カスタマエンジニアの処理

【図１２】

従来例の説明図（その１）

A：半導体ディスク装置のブロック図



B：制御テーブル例

9

	CYLNUM
DRV 0	N1
DRV 1	N2
DRV 2	N3
DRV 3	N4
DRV 4	N5

【図１３】

従来例の説明図（その２）

（シリンダ数を増やす場合）